

Harjoitukset 4

- Harjoitustehtävät palautetaan [ScoreTronicilla](#). Tarjolla on myös [ScoreTronicin käyttöohjeet](#)
- Harjoitukset 4 tulee palauttaa ryhmittäin seuraavasti
 - TTV18S2: **2.10.2019** klo 23.59 mennessä
 - TTV18S3: **1.10.2019** klo 23.59 mennessä
 - TTV18SM: **4.12.2019** klo 23.59 mennessä
- JavaScript-tehtävissä ei saa käyttää jQueryä tai muita vastaavia JS-kirjastoja

Tehtävä 1 [10p] #planone

Luo oheisen mallin ja ohjeistuksen mukainen web-sovellus kiinteistöjen myynti-ilmoitusten esittelemiseksi HTML:n, CSS:n, JavaScriptin ja AJAXin avulla. Ilmoitusten tekstidata tulee olla JSON-muodossa. Tehtävä on vahvasti ohjeistettu ja ohjeita noudattamalla tehtävän vaikeustaso on keskitasoa tälle opintojaksolle. Tästä harjoituksesta tehdään jäljempänä eri tekniikoilla uusia versioita, joten olisi tärkeää tehdä tämä tehtävä pohjatyöksi.

	<p>Suistokatu 3, 00120 , Helsinki</p> <p>5h+kt+2kh+s+wc+khh+vh+p</p> <p><i>Granholmin suunnittelema vuonna 1896 valmistunut Kivipalatsi Pyyntikorttelissa kuuluu tänäkin päivänä Helsingin kauneimpiin rakennuksiin.</i></p> <p>2 904 000 euroa</p>
	<p>Telakkakatu 3 A, 00150, Helsinki</p> <p>4h+k+halli+2xkh+wc+p</p> <p><i>Keittiöstä pariovet avaavat parvekkeen osaksi huonetilaa. Kaksi tilavaa kylpyhuonetta, toisessa amme. Joustavasti muunneltavia huonejakoja, paljon ikkunoita.</i></p> <p>1 044 000 euroa</p>

1 JSON-tiedoston muodostaminen

- Muodosta JSON-tiedosto seuraavasta asuntodatasta: [talotiedot.txt](#)
- Tarkista JSON-tiedoston muoto esim. palvelussa: <http://jsonformatter.curiousconcept.com>

- Tässä luotava `.json`-tunnisteinen tiedosto tallennetaan samaan kansioon jatkossa luotavien HTML5-, CSS- ja JavaScript-tiedostojen kanssa
- Vihjeet
 - JSON-tiedostossa `talot` on ylin objekti
 - `talot`-objekti sisältää `talo`-objekteja taulukossa
 - `talo`-objekti sisältää tiedot "kuva", "osoite", "kuvaus" ja "hinta".
 - Alla mallia tiedoston alusta

```
{
  "talot":
  [
    {
      "kuva": "kuvat/talo1.jpg",
      "osoite": "Suistokatu 3, 00120 , Helsinki",
      "koko": "5h+kt+2kh+s+wc+khh+vh+p",
      "kuvaus": "Granholmin suunnittelema vuonna 1896 valmistunut Kivipalatsi...",
      "hinta": "2 904 000 euroa"
    },
    ...
  ]
}
```

2 Kuvat

- Halutessasi voit käyttää näitä [kuvia](#)
- Kopioi kuvat esim. `kuvat`-nimiseen alikansioon

3 Web-sivun pohja

- Muodosta uusi HTML5-dokumentti alla olevien tietojen mukaisesti:
 - lisää body-elementin sisälle yksi div-elementti, jonka yksilöllinen id on "talot"
 - kutsu `loadJSON()` -funktioita body-elementin onload-tapahtumasta

4 Web-sivun tyylit

- Laadi alla olevien ohjeiden mukaiset tyylimäärittelyt web-sivun elementeille ulkoiseen tiedostoon
- osa elementeistä ohjelmoidaan JavaScriptillä dynaamisesti sivulle (ne eivät ole valmiina HTML5-dokumentissa)
- katso mallia tarvittaessa myös esimerkkikuvasta

Tyylit kaikki talot sisältävälle DIV-elementeille

```
#talot {
  leveys 600px
  korkeus 800px
  tasaus vasemmalle
  ylivuoto arvoon auto
  margin 10px
}
```

Tyylit yhden talon DIV-elementille

```
.taloContainer {  
  display arvo block  
  border 1px solid ja harmahtava väri  
  leveys 570px  
  korkeus 165px  
  valitse mieli fonttisi  
  valitse fontin kooksi 12px  
  margin ja padding 2px  
}
```

Tyylit talon kuvalle

```
.taloImage {  
  leveys 225px  
  korkeus 160px  
  sijoittelu vasemmalle  
  margin 2px  
}
```

Tyylit talon otsikkoteksteille

```
p.otsikko {  
  fonttikoko 14px  
  fontin muoto bold  
}
```

Talon kuvaustekstin tyylit

```
p.kuvaus {  
  fontin tyyli italic  
  fontin koko 10px  
}
```

5 Web-sivun ohjelmointiosuudet

- Ohjelmoi (enimmäkseen kopioi) toiminnot JSON-tiedoston lataamiseen jatkossa annettujen ohjeiden mukaan.
- Luo erillinen .js-tunnisteinen JavaScript-tiedosto haluamallasi nimellä
- Ota käyttöön erillinen JavaScript-tiedosto käyttöön aiemmin luomallasi web-sivulla

6 JSON-tiedoston lataaminen palvelimelta

- `loadJSON()` -funktio käyttää myöhemmin tehtävää `ajax()`-funktia lataamaan json-tiedoston palvelimelta
- Latauksen jälkeen funktio käy JSON-objektin läpi ja käyttää `naytaTalot()` -funktia näyttämään talon tiedot web-sivulla

```
// talot-aulukkoon ladataan talojen tiedot palvelimelta  
var talot = new Array();  
  
// funktiota kutsutaan web-sivulta bodyn onload-tapahtumasta
```

```
function loadJSON() {
    ajax("talotiedot.json", function(response) {
        //console.log("response = " + response);
        // create a json object
        var JSONObject = JSON.parse(response);
        talot = JSONObject.talot;
        //console.log(talot);
        for (var i=0;i<talot.length;i++) {
            naytaTalo(i);
        }
    });
}
```

- Oheinen `ajax()` -funktio käyttää `XMLHttpRequest`-objektia muodostamaan yhteyden web-sivulta palvelimelle
- `XMLHttpRequest`-objektin `open()` -funktio muodostaa yhteyden palvelimelle ja `send()` -funktio lähettää latauspyynnön
- Latauksen tulokset käsitellään `loadJSON()` -funktiossa

```
function ajax(url, fn) {
    var req;
    if (window.XMLHttpRequest) {
        req = new XMLHttpRequest();
    } else {
        req = new ActiveXObject('Microsoft.XMLHTTP');
    }
    req.onreadystatechange = function() {
        if (req.readyState == 4 && req.status == 200) {
            fn(req.responseText);
        }
    }
    req.open('GET', url, true);
    req.send();
}
```

7 Talotietojen näyttäminen web-sivulla

- `naytaTalo()` -funktioita kutsutaan jokaisen talon tietojen näyttämiseksi
- `naytaTalo()` -funktio luo uuden DIV-elementin ja sen sisälle IMG- ja P-elementtejä näyttämään ko. talon tietoja
- Huom: Elementin `setAttribute`-attribuuttia käytetään sitomaan luotuun elementtiin tyylimääreitä

```
function naytaTalo(index) {
    // uusi div
    var taloDiv = document.createElement("div");
    taloDiv.setAttribute("class", "taloContainer");
    var img = document.createElement("img");
    img.setAttribute("class", "taloImage");
    img.setAttribute("src", talot[index].kuva);
    //console.log("kuva="+talot[index].kuva);
    taloDiv.appendChild(img);

    var p1 = document.createElement("p");
    p1.setAttribute("class", "otsikko");
    var text = document.createTextNode(talot[index].osoite);
    p1.appendChild(text);

    var p2 = document.createElement("p");
    var text = document.createTextNode(talot[index].koko);
    p2.appendChild(text);
}
```

```

var p3 = document.createElement("p");
p3.setAttribute("class", "kuvaus");
var text = document.createTextNode(talot[index].kuvaus);
p3.appendChild(text);

var p4 = document.createElement("p");
var text = document.createTextNode(talot[index].hinta);
p4.appendChild(text);

taloDiv.appendChild(p1);
taloDiv.appendChild(p2);
taloDiv.appendChild(p3);
taloDiv.appendChild(p4);

// talot div
var talotDiv = document.getElementById("talot");

talotDiv.appendChild(taloDiv);
}

```

8 Testaa

Palautus

Tehtävä palautetaan osoittamalla URL toimivaan ohjelmaan

Tehtävä 2 [10p]

Alkuhuomautus: Tämän tehtävän vaikeustaso on "vaativa" tälle opintojaksolle.

PHP-ohjelma `ajax-suggest.php` [\[lähdekoodi\]](#) tarjoaa pienen etunimitietokannan. Ohjelma tulostaa sellaiset etunimet tabulaattorein toisistaan erotettuina, joiden **alkuosa** vastaa ohjelman URLin **q**-parametrin arvoa. Esimerkiksi osoitteella `https://student.labranet.jamk.fi/~H1234/ajax-suggest.php?q=a` ohjelma tulostaisi kaikki a:lla alkavat nimet esim.

```
Anni\tAnna\tAnni\tAntti\tAri
```

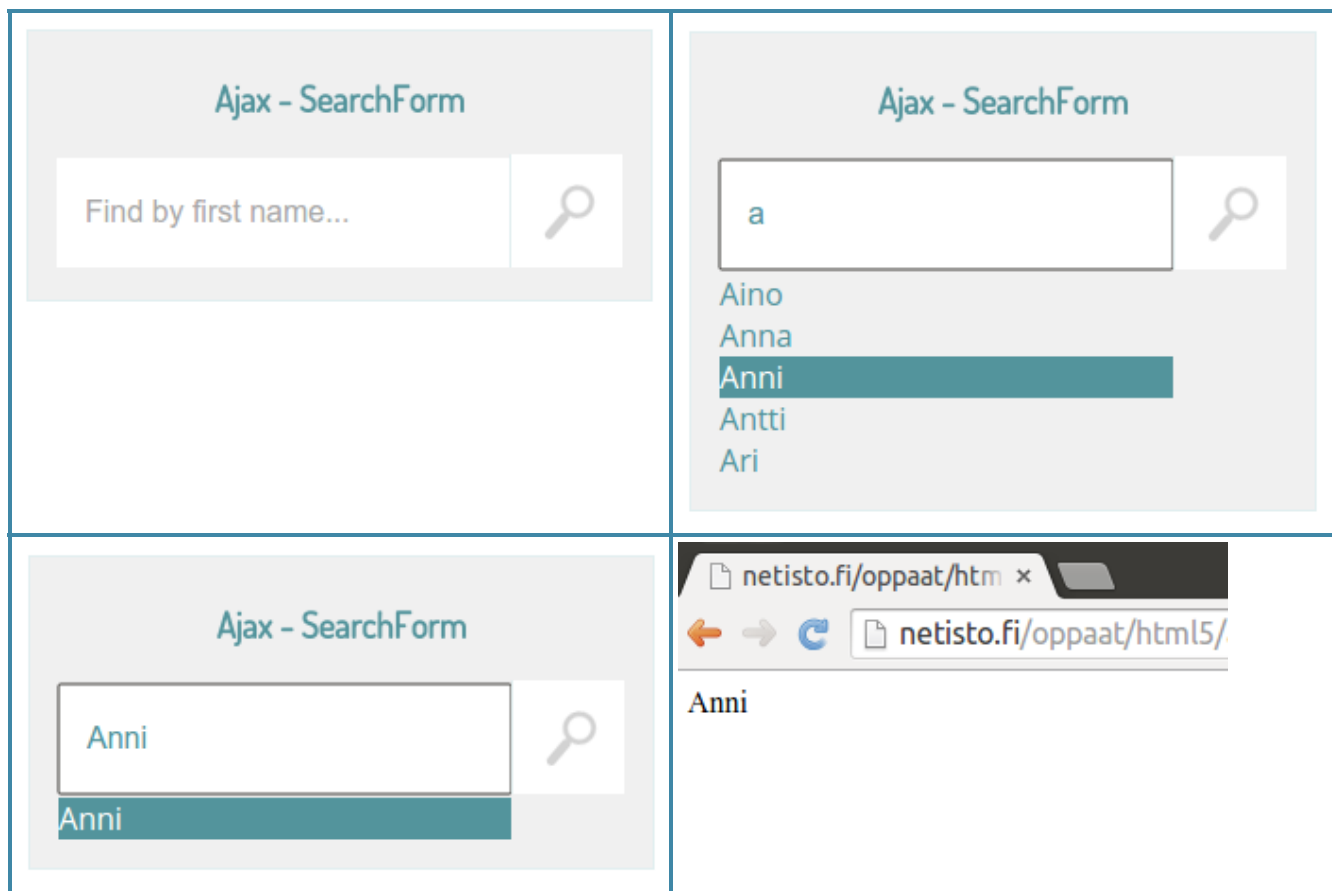
Toteuta `ajax-suggest.php`-ohjelmalle käyttöliittymäksi HTML:llä, CSS:llä, **JavaScriptillä** ja **AJAXilla** oheisen kuvasarjan mukainen etunimien hakukenttä. Tämän tehtävän ratkaisussa ei saa käyttää ulkoisia kirjastoja kuten esim. JQueryä. Tästä tehtävästä maksimipisteet ovat 10 pistettä, jotka voit kerätä valintasi mukaan seuraavista kohdista:

1. [5p] Käyttäjän jokaisen näppäinpainalluksen jälkeen (`onkeyup`) JavaScript/AJAX-toteutuksesi näyttää palvelimen `ajax-suggest.php`-skriptin vastauksen perusteella hakukentän alapuolella sellaiset nimet, joiden alkuosa vastaa hakukentässä olevaan tekstiin: esim. kuvassa kaikki a:lla alkavat nimet.
2. [2p] Hakutulosten ollessa näkyvillä käyttäjän on voitava selata nimiä `nuoli alas` ja `nuoli ylös` -näppäimin. Valittuna oleva kohta tulee olla jotenkin

korostettuna kuten kuvassa nimi *Anni*.

3. [2p] Käyttäjän klikatessa hiirellä jotakin hakutulosta kutsutaan `ajax-suggest.php`-skriptiä muodossa `ajax-suggest.php?q=etunimi` jolloin skripti näyttää klikatun nimen yksinkertaisesti tekstinä oikean alakuvan tapaan.
4. [1p] Käyttäjän painaessa ENTERiä valitun hakutuloksen tulee kopioitua hakukenttään kuten vasemmassa alakuvassa.
5. [1p] Käyttäjän painaessa ESCiä hakukenttä palautetaan alkutilaansa ja hakutulosten lista tyhjätyään.
6. [1p] Hakuikonia (suurennuslasi) tai vastaavaa painiketta klikatessa kutsutaan `ajax-suggest.php`-skriptiä muodossa `ajax-suggest.php?q=etunimi` jolloin skripti näyttää hakukentässä olevan *etunimen* yksinkertaisesti tekstinä oikean alakuvan tapaan.
7. Käyttöliittymän ulkoasun voit päättää itse

Vinkkejä tehtävään 2 [vinkit-h04t02.txt](#). Tehtävää ei ole pakotettu tekemään vinkkien pohjalte.



Palautus

Tehtävä palautetaan osoittamalla URL toimivaan ohjelmaan

Jätetty tarkoituksella tyhjäksi